YMCA
—Why Markov Chain Algebra?—

Mario Bravetti

Dipartimento di Scienze dell'Informazione, Università di Bologna,
Mura Anteo Zamboni 7, I-40127 Bologna, Italy.
E-mail: bravetti@cs.unibo.it

Holger Hermanns

Dependable Systems & Software, Department of Computer Science,
Saarland University, 66123 Saarbruecken, Germany; and
University of Twente, Formal Methods and Tools
P.O. Box 217, 7500 AE Enschede, The Netherlands.
E-mail: hermanns@cs.uni-sb.de

Joost-Pieter Katoen

RWTH Aachen, Software Modeling and Verification,
Ahornstrasse 55, D-52074, Aachen, Germany; and
University of Twente, Formal Methods and Tools,
P.O. Box 217, 7500 AE Enschede, The Netherlands.
E-mail: katoen@cs.rwth-aachen.de

Abstract

Markov chains are widely used to determine system performance and reliability characteristics. The vast majority of applications considers continuous-time Markov chains (CTMCs). This note motivates how concurrency theory can be *extended* (as opposed to *twisted*) to CTMCs. We provide the core motivation for the algebraic setup of *Interactive Markov Chains*. Therefore, this note should have better been baptized YIMC.

Continuous-time Markov chains (CTMCs) are a widely used performance evaluation model. They can be considered as labeled transition systems, where the transition labels—rates of negative exponential distributions—indicate the speed of the system evolving from one state to another. Numerical algorithms allow the computation of important characteristics of a given CTMC with relative ease and comfortable run times, and in a quantifiably precise manner. Using probabilistic model checking techniques

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: `www.elsevier.nl/locate/entcs`

also logical properties can be checked. Several software tools are available to support the specification, numerical analysis, and model checking of CTMCs. This note is not concerned with the analysis, but with the integration of CTMCs in the process algebraic framework for the modeling and analysis of reactive systems.

Interactive Markov chain algebra (IMC) is an extension of classical process algebra in which random delays can be described. Any such delay is specified by a negative exponential distribution. The basic concept is to *add* a delay prefix to process algebra. This simple extension—a clear separation between delays and actions—yields a specification formalism for describing CTMCs in a precise and modular way, resembling the hierarchical nature of typical modern systems. The theory of IMC has been driven by a set of design rationales, which we briefly discuss in the sequel.

IMC is a simple extension of process algebra.

It extends traditional process algebra by a single operator, $(\lambda)\,.\,P$, where λ is an arbitrary positive real value, and $.$ the prefix operator, and P a process algebra term. Intuitively, $(\lambda)\,.\,P$ delays for a time which is exponentially distributed with rate λ prior to exhibiting the behavior of P. Stated differently, the probability to behave like P within t time units is $1 - e^{-\lambda \cdot t}$, or simpler: it takes on average $\frac{1}{\lambda}$ time units to evolve into P.

IMC extends process algebra in a *conservative* way, i.e., the meaning of established composition operators does not change.

IMC is a conservative extension because the operational semantics, equivalence relations, and equational theory remain unaltered for the basic process algebra fragment. Whether one takes CCS, π-calculus, CSP, ACP, μ-CRL or ... as a basis does not make a difference. We took LOTOS, where the basic fragment looks like this:

$$P ::= a\,.\,P \;\Big|\; P + P \;\Big|\; X \;\Big|\; recX.P \;\Big|\; P\,||_S\,P$$

where a is an action, S a set of actions, and $+$ and $||_S$ are the standard choice and parallel composition. Note that the last rationale can also be formulated as "standard process algebra is included in IMC".

IMC encompasses the *algebra of CTMCs*, where bisimulation coincides with lumpability.

The algebra of CTMCs is a fragment of IMC orthogonal to the (standard) process algebra fragment, and is characterized by the following equational laws:

$$\text{(B1)} \quad P+Q = Q+P \qquad \text{(B2)} \quad (P+Q)+R = P+(Q+R)$$

$$\text{(B3)} \quad P+\mathbf{0} = P \qquad \text{(B4)} \quad (\lambda+\mu)\,.\,P = (\lambda)\,.\,P+(\mu)\,.\,P$$

The axioms (B1) through (B3) are well known and standard for process algebra. Axiom (B4) is a distinguishing law and can be regarded as a replacement in the Markovian setting of the traditional idempotency axiom for choice ($P + P = P$). It reflects that the resolution of choice is modeled by the minimum of (statistically independent) exponential distributions. Together with standard laws for handling recursion on classical process calculi, these axioms can be shown to form a sound and complete axiomatization of the CTMC fragment given by:

$$P ::= (\lambda)\,.\,P \;\Big|\; P+P \;\Big|\; X \;\Big|\; recX.P \;\Big|\; P\,||_{\varnothing}\,P$$

$$\boxed{\text{IMC naturally supports phase-type distributions.}}$$

Phase-type distributions can be considered as matrix generalizations of exponential distributions, and include frequently used distributions such as Erlang, Cox, hyper- and hypo-exponential distributions. Intuitively, a phase-type distribution can be considered as a CTMC with a single absorbing state (a state that is never left once reached). The time until absorption of this absorbing CTMC determines the phase-type distribution [9]. In terms of IMC, phase-type distributions can be encoded by explicitly specifying the structure of the CTMC using summation, recursion, and termination ($\mathbf{0}$), as in the IMC-term $\tilde{Q}$ given by $(\lambda)\,.\,recX.(\mu)\,.\,(\mu)\,.\,X + (\lambda)\,.\,\mathbf{0}$. The possibility of specifying phase-type distributions is of significant interest, since phase-type distributions can approximate arbitrary distributions arbitrarily close [9] (i.e., it is a dense subset of the set of continuous distributions). In other words, IMC can be used to express arbitrary distributions, by choosing the appropriate absorbing Markov chain, and (mechanically) encoding it in IMC.

IMC supports *constraint-oriented* specification of random time constraints.

(The term constraint-oriented was coined in [11].) This property enables to enrich existing untimed specifications with random timing constraints by just composition. The description of time constraints can thus take place in a *modular* way, that is, as separated processes that are constraining the behavior by running in parallel with an untimed (or otherwise time-constrained) process. This is facilitated by an *elapse* operator [6] which is used to impose phase-type distributed time constraints on specific actions. The semantics of this operator is defined by means of a translation into the basic operators of IMC—it is, in fact, just "syntactic sugar". Due to the compositional properties of IMC, important properties (e.g., congruence results) carry directly over to this operator. Delays are imposed as time constraints between two actions, and a delay may be "interrupted" if some action of some kind occurs in the meanwhile. That is, the elapse operator is an operator with four parameters, syntactically denoted by

$$[\mathbf{on}\ S\ \mathbf{delay}\ D\ \mathbf{by}\ Q\ \mathbf{unless}\ B]:$$

- a phase-type distribution Q that determines the duration of the time constraint,
- a set of actions S (start) that determines when the delay (governed by Q) starts,
- a set of actions D (delay) which have to be delayed, and
- a set of actions B (break) which may interrupt the delay.

Thus, for instance, $[\mathbf{on}\ \{a\}\ \mathbf{delay}\ \{b\}\ \mathbf{by}\ \tilde{Q}\ \mathbf{unless}\ \varnothing]$ imposes the delay of $\tilde{Q}$ (modeling a phase-type distribution) between a and b. Semantically, the intuition behind this operator is that it enriches the chain Q with some synchronization potential that is used to initialize and reset the time constraint in an appropriate way. The time constraint is imposed on a process P by means of parallel composition, such as in

$$P\,||_{S\cup D\cup B}\,[\mathbf{on}\ S\ \mathbf{delay}\ D\ \mathbf{by}\ Q\ \mathbf{unless}\ B].$$

IMC adds nondeterminism and interaction to CTMCs.

Interactive Markov chains can be used to specify CTMCs, but due to the presence of nondeterminism (inherited from standard process algebra), the model underlying IMC is richer. In fact, it is the class of continuous-time Markov decision chains [10], a strict superset of CTMCs. Nondeterminism is one of the vital ingredients of process algebra and hence of IMC.

IMC has a well-understood equational theory.

See [4, Chaper 5] for sound and complete equational characterizations of strong and weak bisimilarity for the full calculus—including recursion and open terms.

Conclusion

In a nutshell, a well thought-out choice of basic algebraic operators makes IMC a calculus with a unique set of distinguishing properties. The theory of IMC is developed in [4,5]; see also [1]. It is worth noticing that calculi like PEPA [7] or $\mathrm{EMPA}_{\mathrm{gr}}$ [2] do not possess all of the aforementioned properties. In particular, they are not conservative extensions of process algebra as delays and actions are twisted rather than separated. The separation of delays and actions allows to treat action synchronization as in standard process algebra and is also one of the key principles to obtain process algebraic frameworks for more general distributions [3,8].

References

[1] M. Bravetti. Revisiting Interactive Markov Chains. *Electr. Notes Theor. Comput. Sci.*, 68(5), 2002.

[2] M. Bravetti and M. Bernardo. Compositional asymmetric cooperations for process algebras with probabilities, priorities, and time. *Electr. Notes Theor. Comput. Sci.* 39(3), 2000.

[3] M. Bravetti, R. Gorrieri. The theory of interactive generalized semi-Markov processes. *Theor. Comput. Sci.* 282(1): 5-32, 2002.

[4] H. Hermanns. *Interactive Markov Chains and the Quest for Quantified Quality.* LNCS 2428, 2002.

[5] H. Hermanns, U. Herzog, and J.-P. Katoen. Process algebra for performance evaluation. *Theor. Comput. Sci.*, 274 (1-2):43 - 86, 2001.

[6] H. Hermanns and J.-P. Katoen. Automated compositional Markov chain generation for a plain-old telephony system. *Science of Comput. Prog.*, 36(1):97–127, 2000.

[7] J. Hillston. *A Compositional Approach to Performance Modelling.* Cambridge University Press, 1996.

[8] P.R. D'Argenio, J.-P. Katoen and E. Brinksma. An algebraic approach to the specification of stochastic systems (extended abstract). In *Programming Concepts and Methods.* Chapman & Hall, pp. 126–147, 1998.

[9] M.F. Neuts. *Matrix-geometric Solutions in Stochastic Models–An Algorithmic Approach.* The Johns Hopkins University Press, 1981.

[10] M.L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley & Sons, 1994.

[11] C.A. Vissers, G. Scollo, M. van Sinderen and E. Brinksma. On the use of specification styles in the design of distributed systems. *Theor. Comput. Sci.*, 89(1):179–206, 1991.